

KSC: Knowledge, Structure, and Constraint — A Functional Framework for Human–AI Differences and Collaboration

Liu Ming

Version V3.3.1 · August 2026

Abstract

Current AI systems can draw on broad bodies of knowledge and rapidly generate complex structures, yet their capabilities still vary substantially across tasks. Some problems are constrained mainly by the information available; some require new problem representations, relationships, or solution structures to be formed; and some tasks, despite high-quality local generation, gradually lose goals, premises, boundaries, or acceptance criteria across long chains of work involving many conditions and strong state dependencies. Describing these differences along a single axis of “more or less intelligent” is insufficient for explaining the capability boundaries of current AI or the forms of human–AI collaboration that arise in practice.

This paper proposes the Knowledge–Structure–Constraint Framework (KSC), which describes intelligent activity in complex goal-directed tasks using three progressively dependent functional coordinates. K denotes the information and state currently accessible to or usable by the system; S denotes the generation of problem representations, relationships, solutions, and action paths on the basis of K; and C denotes the continuing constraining effect of established goals, premises, boundaries, invariants, and acceptance criteria on S and on the state transitions it induces. KSC is intended as a low-dimensional coordinate system for understanding the structure of intelligent capability, not as a diagnostic tool that requires concrete behaviors or failures to be assigned exclusively to K, S, or C.

The paper further examines the role of C in finite reasoning and generation processes. Every actual model call and its internal reasoning process has practical limits, so complex work often has to span multiple calls, reasoning stages, and state transitions. Constraint adherence within a single call and consistency maintenance across multiple calls can be understood as manifestations of C at different timescales: a finite reasoning process must not only form new structures, but also keep established goals, premises, and boundaries operative in subsequent generation. The fact that constraint information has entered K does not mean that C is automatically realized; C exists only when those conditions actually constrain reasoning, action, and state transitions. This is a functional relationship and does not presuppose an independent C module, a fixed partition of resources, or any particular search mechanism.

On the basis of KSC, the paper advances a provisional comparative claim. AI systems centered on current large language models have clear advantages in the breadth of knowledge they can access and in the speed, scale, and iterative capacity with which they generate candidate structures. Typical human work systems, by contrast, operate continuously in real environments, bear responsibility and the consequences of action, and make extensive use of documents, institutions, tests, permissions, and state records. In many complex tasks, these conditions usually provide humans with more stable means of maintaining constraints. In engineering tasks where goals, boundaries, and acceptance criteria are relatively explicit, the main additional contribution from humans often lies in C; in open-ended problems where the problem representation and direction of exploration have not yet been established, humans may also participate directly in S. Humans, AI, and external systems can jointly realize the functions represented by K, S, and C, producing different structures of human–AI collaboration. KSC does not attempt to provide a complete

account of consciousness, agency, the origins of values, or the underlying computational mechanisms of models.

Keywords: KSC framework; human–AI collaboration; large language models; knowledge base; structure generation; constraint maintenance

1. Introduction

The use of generative AI is expanding from single-turn question answering and local content generation to longer-horizon, more complex work in software development, research, planning, and agentic systems. Models also increasingly operate not as isolated components but as parts of complete task systems that combine retrieval, tools, external state, verification mechanisms, and human judgment. As a result, the gap between “what a model can generate in one attempt” and “what a complete system can reliably carry through over time” is becoming an important question for understanding AI capability.

Existing discussions commonly explain these differences in terms of knowledge deficits, reasoning ability, context utilization, instruction following, external memory, long-horizon planning, or human–AI complementarity. These research directions correspond to different technical mechanisms, but they lack a sufficiently simple functional language for placing them within a common capability structure. Consequently, terms such as “model capability” or “intelligence level” often simultaneously refer to knowledge, generation, control, and system-level coordination, making stable comparison across models, tasks, and human–AI systems difficult.

This paper therefore focuses on three related questions. Can a low-dimensional capability coordinate system provide a common language for describing different kinds of intelligent function in complex tasks? When a task exceeds what can be completed within one finite reasoning and generation process, how do established goals, premises, boundaries, and acceptance criteria continue to shape later processes? And how can these capabilities be distributed across humans, AI, and external systems to produce different forms of human–AI collaboration?

The discussion focuses primarily on complex tasks in which goals and evaluation criteria have already been established, or jointly confirmed through collaboration, and in which the objective is to produce a deliverable result or change an external state.

2. KSC: A Three-Level Functional Framework of Knowledge, Structure, and Constraint

2.1 Overall Structure

KSC divides information-processing activity in complex goal-directed tasks into three functional levels linked by a progressive dependency relationship:

Functional order	KSC function	Core question	Typical objects or manifestations
First-order: available information	K: Knowledge Base	What can the system currently obtain and use?	Parametric knowledge, long-term memory, context, retrieval results, situational facts, task state, and experience records

Functional order	KSC function	Core question	Typical objects or manifestations
Second-order: structure generation	S: Structure Generation	How does the system form new structures from available information?	Abstraction, classification, relationship discovery, problem representation, reasoning, decomposition, planning, solution generation, and corrective paths
Third-order: generation constraint	C: Constraint Maintenance	What continues to constrain generation and state transitions?	The continuing control exerted by goals, premises, boundaries, permissions, invariants, and acceptance criteria over generation, action, and state updates

The basic relationship among the three is as follows: K provides material for S; S organizes available information into new problem structures, relationships, and candidate actions; and C uses conditions that have already been established as valid to accept, restrict, revise, roll back, authorize updates to, or validate S and the state transitions it produces.

Here, the first-, second-, and third-order distinction refers to a functional dependency relationship rather than a fixed temporal sequence, three physically independent modules, or higher-order variables in the mathematical sense. The functions represented by K, S, and C can all be carried out within the same model call, by the same human actor, or by the same external system. An evaluation result can become part of K at a later stage; when reasoning is used to judge an outcome against established conditions, the same reasoning process can serve the C function; and when it generates a new corrective solution, it can simultaneously serve the S function.

KSC is therefore a capability coordinate system rather than a component taxonomy. Its concern is what functional role information and computation play in the current task, and what capability conditions the complete system possesses along each of the three functional dimensions.

2.2 K: Knowledge Base

K is the information and state that, at the current stage of a task, have entered the range that the system can access or use and are, in principle, available for subsequent processing. It includes knowledge consolidated in model parameters or human long-term memory, as well as context, external retrieval, task documents, situational facts, historical decisions, and current state.

The mere objective existence of information does not mean that it has become usable K for the current task. Information enters the task's knowledge base only when the system can access or use it at the relevant stage. K by itself does not guarantee that information will be correctly recognized, appropriately weighted, or actually used in subsequent generation. The significance of K therefore lies not only in the total amount of information available, but also in whether relevant information can enter the usable range of the current task.

A rich K does not automatically imply an ability to solve new problems. Existing information is

only the material for subsequent structure generation; when new relationships, problem representations, or action paths are required, the system still needs S.

2.3 S: Structure Generation

S is the function of forming new structures on the basis of K. It includes problem representation, relationship discovery, abstraction, classification, causal reasoning, task decomposition, planning, solution generation, formation of search directions, and the generation of new corrective paths in response to feedback.

S does not merely mean producing an answer. It also includes changing how a problem is represented, discovering new variables and relationships, proposing candidate explanations or technical paths, and reorganizing existing information after receiving feedback. A system may possess a large amount of K yet fail to make progress because a crucial structure is missing; conversely, a new instance of S may turn previously scattered information into an actionable problem structure.

The outputs of S can become new K in later stages and can also change the problem space that the system is able to perceive next. Structure generation by itself does not guarantee that newly generated structures will continue to satisfy established goals, premises, and boundaries. That continuing constraining function is described by C.

2.4 C: Constraint Maintenance

C, or Constraint Maintenance, is the function by which already established goals, premises, boundaries, priorities, invariants, permissions, and acceptance criteria continue to exert constraints across ongoing reasoning, action, and state transitions, and can be updated or lifted only through authorization or explicit confirmation.

When goals, boundaries, and acceptance criteria are recorded, stored, and made accessible, they belong to K as information. When that information actually affects the selection of information, candidate structures, permission to act, evaluation of results, state updates, or judgments of completion, it performs the function of C. C is therefore not a special category of information independent of K, but a functional relationship in which existing conditions continue to constrain generation and state transitions.

C is also not equivalent to simply remembering requirements. The fact that a constraint has been written into a prompt, document, memory store, or state does not mean that it has taken effect. At the relevant stage, the system must still recognize its validity, scope, priority, and authority for modification, and use it to restrict, filter, revise, roll back, or terminate generation paths that do not satisfy the conditions.

The distinction between S and C is not an exclusive classification of low-level computational operations but a distinction between functional roles: S concerns what new structures the system forms; C concerns whether already established conditions continue to determine which structures, actions, and state transitions may be accepted. The concrete process that serves the C function may itself include reasoning, comparison, classification, and other computations that can also appear in S, and it may simultaneously generate new structures through S.

Determining whether a current structure violates an established condition or requires revision or roll-back can serve a C function; forming a concrete corrective solution and a new action path serves an S function. When existing conditions are insufficient to determine a problem representation or direction of exploration and a new abstraction, relationship, or candidate direction must be formed, the activity still belongs to S. The same complex behavior may therefore perform both S and C without having to be assigned exclusively to either one.

Constraints are not permanently fixed. New facts can enter K, and new structure-generation activity can produce a revised understanding of the task; after authorization or explicit confirmation, goals, pri-

orities, and boundaries may be updated. A recognized and authorized change is a normal update within C. Unrecognized forgetting, weakening, overwriting, or unauthorized modification constitutes constraint failure or drift.

2.5 Level of Analysis and Boundaries

KSC describes functional relationships in complex tasks rather than an exclusive taxonomy of failures. The same behavior, tool, or failure may involve K, S, and C simultaneously, or may serve different functions at different stages. The framework does not require every practical problem to be uniquely mapped to one letter, nor is it intended to provide a project-level fault-diagnosis procedure.

KSC instead establishes a higher-level capability coordinate system: what kind of information base a system can obtain, what kinds of new structures it can form, and over what scope established goals, premises, and boundaries can continue to constrain task progression. This coordinate system can be used to compare different models, human actors, and hybrid systems, and to describe how the same system exhibits different capability profiles under different task structures.

3. Current Human–AI Capability Differences Through the KSC Lens

3.1 System-Level Comparison

KSC does not compare the human brain and a model as if they possessed isomorphic low-level capabilities. It compares the levels a complete task system can achieve along K, S, and C, and examines which actors and mechanisms jointly provide those capabilities. Humans can draw on long-term memory, embodied environments, tools, institutions, and social relationships; AI systems can draw on models, context, retrieval, memory, executors, and verification mechanisms.

Differences in implementation therefore do not prevent functional comparison. What matters is the range over which a system can obtain relevant information, the kinds of new structures it can form, and the scale and duration of tasks over which it can keep established goals and boundaries operative.

3.2 Characteristics of K and S in Current AI

Current large language models possess broad parametric knowledge and can also connect to retrieval systems, tools, and external documents. At relatively low cost, they can rapidly perform classification, summarization, comparison, decomposition, planning, explanation, writing, and code generation, substantially increasing the speed, quantity, and iterative capacity of knowledge use and candidate structure generation within individual tasks.

A preregistered experiment involving 758 knowledge workers found that GPT-4 assistance increased the number of tasks completed, speed, and quality when tasks fell within the model’s capability frontier; on a complex management task outside that frontier, however, AI assistance could also reduce accuracy [1]. The result indicates that the benefits of AI assistance remain constrained by task structure and the model’s capability boundary. From a KSC perspective, it can be interpreted as showing that advantages in knowledge access and candidate structure generation can substantially improve local task performance but do not by themselves guarantee overall performance once the task exceeds the model’s capability boundary.

The S advantage described here refers primarily to current AI’s advantage in the speed, scale, and iterative capacity of candidate structure generation. It does not imply that AI can independently discover the correct problem representation or direction of exploration in every open-ended problem. When a decisive structure has not yet been formed, humans may still provide crucial S.

3.3 A Provisional Difference in C Between Typical Human Work Systems and Current AI Systems

Typical human work systems are also constrained by individual knowledge coverage, structure-generation speed, and working-memory limits. However, humans normally remain situated in continuous real-world environments, are constrained by responsibility, social evaluation, risk, and the consequences of action, and continuously use documents, institutions, tests, permissions, and state records to maintain task continuity. In many real-world complex tasks, these conditions usually give typical human work systems more mechanisms through which C can remain effective.

An engineer may not generate the optimal architecture in one attempt, but will usually continue to know that the production environment must not be broken, failing tests must not be ignored, and user data must not be deleted arbitrarily. A manager may be unable to formalize every variable, yet remains continuously constrained by resources, time, responsibility, and real-world feedback. In many cases, the reliability advantage of a typical human work system over an isolated model call does not come from faster structure generation, but from the persistent binding of goals and boundaries to the real environment.

This comparison is not a claim of permanent human advantage. Working memory and cognitive control themselves have capacity and stability limits [2, 3], and humans also depend heavily on external tools and organizational mechanisms to accomplish long-horizon complex work. As internal model control, persistent state, verification, and external governance mechanisms develop, the amount of C that complete AI systems can provide will continue to expand.

3.4 Task Structure Determines How Human and AI Advantages Combine

In engineering tasks where goals, boundaries, and acceptance criteria are relatively explicit, current AI can often already perform much of the required K and S. The main additional contribution from humans therefore tends to lie in C—for example, maintaining goals, controlling boundaries, preserving state continuity, and carrying out real-world acceptance checks.

In more open-ended problems, the problem representation, mode of abstraction, and effective direction of exploration may themselves remain undetermined. Human contributions can then extend from C to direct participation in S. Human–AI division of labor is therefore not a fixed role allocation, but a task-dependent combination of capability conditions across K, S, and C.

The value of KSC lies in providing a more general coordinate system than formulations such as “humans think, AI executes” or “AI generates, humans supervise,” allowing these provisional capability differences to be described within one framework.

4. Further Analysis of C: Finite Reasoning Processes and Long-Horizon Task Continuity

4.1 Finite Reasoning and Generation Processes

Every actual intelligent process is limited by time, computation, information access, and the scope of possible action. For current large language models, a single call has practical limits on the context it can accommodate, the reasoning it can perform, the output it can generate, and the extent to which it can interact with an external environment. Even as these limits expand, as long as each individual process remains finite, there will still be tasks that cannot be completed in full within one local generation process.

Complex work must therefore usually be unfolded across multiple local processes. A system may call a model multiple times, organize multiple reasoning, verification, and correction stages within a single call, or repeatedly pass state among models, tools, and humans. Each local process may exhibit strong knowledge access and structure generation, but a long-horizon task additionally requires conditions established

earlier and still in force to continue shaping later processes.

4.2 C Within and Across Model Calls

C is not a capability that appears only after a task spans multiple calls. Even within a single extended model call, goals, premises, and boundaries specified or formed earlier must continue to shape later reasoning and generation. As generation proceeds, the model forms new structures while also needing to select, judge, suppress, or revise them in accordance with existing conditions.

From the functional perspective of KSC, S and C can be realized by the same finite reasoning process. The process performs S when it forms new problem representations, relationships, solutions, and action paths; it performs C when it keeps established goals, premises, and boundaries operative over those generations. The mere presence of constraint information does not mean that C is automatically realized. C is realized only when those conditions continue to affect selection, judgment, permission to act, and state transitions during actual reasoning. This is a claim about functional roles, not a claim that a model contains explicit S/C compute quotas, separable resource partitions, a particular search process, fixed attention partitions, or a dedicated C neural module.

When a task spans multiple calls, the same functional requirement remains. Through context, external state, memory, or other mechanisms, a new call must make conditions that remain valid from previous stages available again as part of current K; those conditions must then actually constrain the new reasoning process in order for C to continue.

Constraint adherence within one call and consistency maintenance across calls can therefore be understood as manifestations of C at different timescales.

4.3 Continuity in Long-Horizon Tasks

As a complex task progresses through multiple local stages, later stages no longer face only the original task description. They face a continuing task state that includes prior decisions, state changes, excluded paths, and progress toward acceptance. If these established and still-valid conditions no longer affect new generation, the system may continue to produce locally reasonable outputs while reopening previously excluded paths, changing boundaries without authorization, ignoring confirmed decisions, or mistaking local completion for completion of the overall task.

In long-horizon tasks, C therefore describes how shared goals, premises, boundaries, and acceptance conditions continue to constrain multiple local generations. C cannot guarantee success: the system still needs sufficient K, S, and capacity to interact with its environment. But local generation quality alone is likewise insufficient to describe whether a long-horizon task can remain coherent.

4.4 Several Levels of Failure from the Perspective of C

From the perspective of C, relevant difficulties do not arise from a single mechanism. Even when goals and constraints have entered the current context, they may still fail to influence generation reliably. Position effects in long-context information use [6], degradation of control performance observed with a Stroop-conflict paradigm [7], and conflicts between parametric and contextual knowledge [8] reveal different limitations in information use and control. None directly proves the existence of a unified C mechanism, but all are adjacent to the problem in which “relevant conditions are available yet fail to exert a stable constraining effect.” Benchmarks of complex instruction following likewise show that current models still have substantial difficulty satisfying constraints in settings that combine multiple constraint types, multiple tasks, and complex compositional requirements [13, 14].

Additional problems arise across calls. In independent calls without persistent task state, interpretations of goals, decisions, boundaries, and acceptance states formed in one call do not automatically

become stable internal state in the next. Long-horizon dialogue and agent systems therefore often use context replay, external memory, or explicit state management to preserve continuity. MemGPT demonstrates one implementation that extends cross-session continuity through hierarchical memory and virtual-context management [9]. External persistence can keep relevant information available, but after that information re-enters the context, the model must still interpret and follow it correctly.

As tasks involve more state transitions, delegation, revision, and intermediate decisions, constraints can also drift through repeated preservation, inheritance, interpretation, and execution. Related work has described the gradual loss of operational effectiveness of constraints along multi-agent trajectories as constraint drift [10]. MasDrift further measures the inheritance of authorization constraints in controlled multi-agent tasks and reports systematic relationships between delegation architecture and hierarchy depth, unauthorized actions, and failures to preserve authorization constraints [11]. These conceptual claims and empirical findings correspond to specific problem settings and should not be generalized directly into a universal law of all long-horizon tasks, but they do indicate that long-horizon constraint maintenance can be studied as a distinct capability problem.

5. Human–AI Collaboration Through the KSC Lens

5.1 Humans, AI, and External Systems Can Jointly Realize K, S, and C

Traditional descriptions of human–AI collaboration often take the form of “humans specify the requirements and AI executes the task” or “AI generates and humans supervise.” These descriptions capture some surface divisions of labor, but they do not adequately represent how human and AI roles vary across tasks.

From the KSC perspective, K, S, and C in a complex task can be jointly provided by humans, AI, and external systems. Humans can contribute situational knowledge, goals, and high-level structural judgments; AI can draw on large bodies of knowledge and rapidly form candidate structures; external systems can preserve facts and state, enforce formal rules, and provide testing and verification. Which actor performs which function depends on task structure and current capability rather than on predetermined role boundaries.

In the engineering tasks emphasized in this paper, one common pattern is that AI already performs much of the required K and S, while humans and the engineering environment carry more of the C function. But when the problem representation, technical direction, or solution path has not yet been formed, humans can participate directly in S. And when external data, state, and retrieval are crucial, tool systems can substantially expand K.

5.2 How External Systems Extend System Capability

External tools can simultaneously alter a system’s information availability, structure generation, and constraint-maintenance capabilities. Documents and retrieval can expand K; planning tools, search, and verification feedback can shape S; state records, permissions, tests, audits, and rollback mechanisms can help goals, boundaries, and acceptance conditions remain effective across multiple stages of a task.

Such mechanisms often change several functional directions at once. The more important question is how the system’s capabilities along K, S, and C change after those mechanisms are added.

Externalization is especially important for C. Humans can manage projects far beyond what can be held in working memory at any one time, not because every goal, boundary, and state remains continuously represented in the brain. Documents, institutions, checklists, version histories, tests, permissions, meeting records, and real-world feedback have long provided part of the continuity needed for complex tasks. Research on situated action and the extended mind likewise suggests that human cognition and action

can be jointly realized through dependence on environments and external tools [4, 5].

AI systems can similarly use task specifications, state records, permissions, state machines, tests, verifiers, audits, and version control so that part of C is carried by persistent external mechanisms. The significance of these mechanisms is not merely that they increase the amount of information a model can see. They also allow previously confirmed goals, boundaries, and acceptance conditions to continue exerting effects across multiple finite reasoning stages.

5.3 A Structural View of Human–AI Collaboration

From the KSC perspective, AI autonomy should not be evaluated solely by the quality of a model’s generation on an isolated task. It also requires examining whether the complete system can obtain the necessary K, form sufficient S, and maintain the required C through the model itself or through external mechanisms. Changes in models, tools, and engineering mechanisms can all alter the system’s capabilities along these three dimensions.

KSC therefore offers a coordinate system for understanding how human and AI capabilities combine and change, rather than prescribing a single workflow for human–AI collaboration.

6. Theoretical Position, Boundaries, and Testable Propositions

6.1 Relationship to Cognitive Theory and Extended Cognition

Psychology has long distinguished crystallized abilities associated with acquired knowledge from fluid abilities involved in discovering relationships and solving novel problems [12]. Research on working memory focuses on the short-term maintenance and manipulation of information [2], while theories of cognitive control emphasize the active maintenance of goals and the means for achieving them, using those representations to guide subsequent thought and action [3].

These theories cover important local functions related to KSC, but KSC does not equate K, S, and C with any three psychological faculties. It analyzes functional relationships in the complete task system. K can reside in model parameters, human long-term memory, context, or external documents; S can be performed by a human, a model, or a tool system; and C can be jointly supported by cognitive control, real-world responsibility, permissions, tests, state machines, and governance processes.

Research on situated action and extended cognition emphasizes the dependence of action on environments and external tools [4, 5]. KSC is compatible with these traditions while adding a functional coordinate system for describing the overall capability structure of systems composed of humans, AI, and external mechanisms.

6.2 Relationship to Adjacent Research on Large Language Models

Research on long-context utilization, knowledge conflict, complex instruction following, external memory, and constraint drift has revealed different limitations of current AI in information use, instruction maintenance, and cross-stage state inheritance [6–11, 13, 14]. These phenomena involve different underlying mechanisms and should not be reduced to context length, insufficient memory, or a single control deficit.

KSC provides a higher-level common descriptive language: what information is available to the system, how it forms new task structures, and how established conditions continue to shape task progression. This makes it possible to compare capability differences identified by these lines of research within the same functional coordinate system without assuming that they share the same underlying mechanism.

6.3 The Theoretical Increment of KSC

The contribution of KSC is not to introduce knowledge, reasoning, cognitive control, or human–AI complementarity for the first time. Its theoretical increment is to organize available information, structure generation, and constraint maintenance into a concise three-level functional coordinate system, expressed as first-, second-, and third-order functions.

This coordinate system makes it possible to discuss “AI capability” as more than a single scalar quantity. The same overall performance may arise from different combinations of K, S, and C, while the same model may exhibit different complete-system capabilities under different external systems and task structures. Knowledge use, problem representation, structure generation, goal maintenance, state continuity, and human–AI collaboration can therefore be discussed within one theoretical language without assuming that they share the same underlying mechanism.

6.4 Framework Boundaries and Empirical Propositions

The first-, second-, and third-order relationship among K, S, and C is a functional division proposed in this paper. Its evaluation should begin with whether the definitions are internally coherent, whether the functional relationships are clear, and whether the framework provides an explanatory common coordinate system for complete tasks. It does not require the three functions to be statistically independent, nor does it require every behavior or failure to be assigned uniquely to one category.

The human–AI comparisons derived from the framework, the specific failure mechanisms of C, and the effects of external systems are empirical claims that can be continuously tested. Future research can examine the following questions:

1. Do humans and AI exhibit stable differences in their relative K, S, and C capability profiles across different task types?
2. As tasks expand from short-horizon local generation to long-horizon complex work, which capability conditions become the dominant limitations?
3. How do external memory, retrieval, planning, state, permissions, testing, and verification mechanisms respectively change the KSC capability profile of the complete system?
4. How does the continuing effectiveness of goals, boundaries, and confirmed decisions change within a single reasoning process and across multiple calls?
5. Can KSC explain differences in the behavior of the same model across task structures and system configurations better than a single aggregate capability score?

These questions are intended to test the empirical claims derived from the framework and the scope of their applicability, rather than to turn KSC into a project-level fault-diagnosis method.

7. Conclusion

This paper proposes KSC, which describes intelligent activity in complex goal-directed tasks using three progressively dependent functional coordinates. K denotes the information and state currently accessible to or usable by the system; S denotes the formation of new problem representations, relationships, solutions, and action paths on the basis of that information; and C denotes how already established goals, premises, boundaries, invariants, and acceptance conditions continue to constrain new generation and state transitions. On this view, “intelligent capability” is no longer represented only by a single scale of greater or lesser capability, but can be understood along the three directions of information base, structure generation, and constraint maintenance.

Within this coordinate system, current differences between humans and AI can be described more clearly. AI systems centered on current large language models have substantial advantages in the breadth of knowledge access and in the speed, scale, and iterative capacity of candidate structure generation. Typi-

cal human work systems, because they operate continuously in real environments and rely on responsibility, feedback, and extensive external mechanisms, usually possess more stable conditions for maintaining constraints in many complex tasks. In engineering tasks, the main additional contribution from humans often lies in C; in open-ended problems, humans may also participate directly in crucial S. These are provisional capability structures, not permanent divisions of labor between humans and AI.

The paper has further examined C in greater depth. Every actual reasoning and generation process is finite, so complex tasks often have to span multiple local processes. Constraint adherence within a single call and consistency maintenance across multiple calls can be understood as manifestations of C at different timescales: established goals, premises, and boundaries must continue to exert a constraining effect on subsequent generation. Internal model control, cross-call state inheritance, and constraint drift along long trajectories illustrate different levels at which this problem may arise.

C also need not be provided entirely by the model itself. Just as humans have long depended on documents, institutions, tests, and real-world feedback, AI systems can use external state, permissions, verification, audits, and rollback mechanisms to extend their long-horizon task capabilities. More generally, humans, AI, and external systems can jointly realize K, S, and C, and the division among them can be recomposed as model and engineering conditions change. KSC therefore offers a foundational coordinate system for understanding the structure of intelligent capability, allowing advances in AI capability, capability limitations, and forms of human–AI collaboration to be discussed in a common functional language.

References

- [1] Dell’Acqua, F., McFowland, E. III, Mollick, E., Lifshitz, H., Kellogg, K. C., Rajendran, S., Krayner, L., Candelon, F., & Lakhani, K. R. (2026). Navigating the jagged technological frontier: Field experimental evidence of the effects of artificial intelligence on knowledge worker productivity and quality. *Organization Science*, 37(2), 403–423. <https://doi.org/10.1287/orsc.2025.21838>
- [2] Baddeley, A. (2012). Working memory: Theories, models, and controversies. *Annual Review of Psychology*, 63, 1–29. <https://doi.org/10.1146/annurev-psych-120710-100422>
- [3] Miller, E. K., & Cohen, J. D. (2001). An integrative theory of prefrontal cortex function. *Annual Review of Neuroscience*, 24, 167–202. <https://doi.org/10.1146/annurev.neuro.24.1.167>
- [4] Suchman, L. A. (1987). *Plans and Situated Actions: The Problem of Human-Machine Communication*. Cambridge University Press.
- [5] Clark, A., & Chalmers, D. J. (1998). The extended mind. *Analysis*, 58(1), 7–19. <https://doi.org/10.1093/analys/58>.
- [6] Liu, N. F., Lin, K., Hewitt, J., et al. (2024). Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12, 157–173. https://doi.org/10.1162/tacl_a_00638
- [7] Patel, S. C., Wang, H., & Fan, J. (2026). Deficient executive control in transformer attention. *PNAS Nexus*, 5(6), pgag149. <https://doi.org/10.1093/pnasnexus/pgag149>
- [8] Xu, R., Qi, Z., Guo, Z., Wang, C., Wang, H., Zhang, Y., & Xu, W. (2024). Knowledge conflicts for LLMs: A survey. *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, 8541–8565. <https://doi.org/10.18653/v1/2024.emnlp-main.486>
- [9] Packer, C., Wooders, S., Lin, K., Fang, V., Patil, S. G., Stoica, I., & Gonzalez, J. E. (2023). MemGPT: Towards LLMs as operating systems. *arXiv preprint arXiv:2310.08560*.
- [10] Li, T., Ma, Y., Wen, H., et al. (2026). Safe multi-agent behavior must be maintained, not merely asserted: Constraint drift in LLM-based multi-agent systems. *arXiv preprint arXiv:2605.10481*.
- [11] Xu, Z., Zhang, X., Luo, H., Jin, Y., Dong, Y., & Salam, H. (2026). MasDrift: Benchmarking authorization preservation across multi-agent architectures. *arXiv preprint arXiv:2608.07556*.

- [12] Horn, J. L., & Cattell, R. B. (1966). Refinement and test of the theory of fluid and crystallized general intelligences. *Journal of Educational Psychology*, 57(5), 253–270. <https://doi.org/10.1037/h0023816>
- [13] Zhang, T., Zhu, C., Shen, Y., et al. (2025). CFBench: A comprehensive constraints-following benchmark for LLMs. *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 32926–32944. <https://doi.org/10.18653/v1/2025.acl-long.1581>
- [14] Zou, T., Zhang, X., Yu, H., Wang, M., Huang, F., & Li, Y. (2025). EIFBENCH: Extremely complex instruction following benchmark for large language models. *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, 20930–20953. <https://doi.org/10.18653/v1/2025.emnlp-main.1059>